CLAIMS

What is claimed is:

1. A computer-implemented method for assisting the detection of critical memory leaks in a software program, the method comprising the steps of:

monitoring an amount of available memory for the software program;

determining if the amount of available memory for the software program is less than a predetermined amount; and

in response to such determination, storing a current stack walkback of each object currently referenced by the software program prior to the available memory dropping below an amount necessary to store the current stack walkback wherein the current stack walkback assists in the detection of a critical memory leak.

2. The method according to claim 1, further comprising the steps of:

monitoring a specified one or more analysis properties of the objects referenced by the software program, wherein the one or more specified analysis properties consists of at least one of an object's age and an object's instance count;

determining if any analysis property of the objects being referenced following a garbage collection process exceeds a predetermined limit for such analysis property, wherein the predetermined limit for an object's age is an object age limit and the predetermined limit for an object's instance count is an object instance count growth value; and

identifying any objects determined to have one or more analysis properties that exceeds that property's predetermined limit.

3. The method according to claim 2, further comprising the step of calculating an object's age by timing a current period starting when the respective object was instantiated.

4. The method according to claim 2, further comprising the step of calculating object instance count growth as the magnitude of growth of an object's instance count over a given time period.

5. The method according to claim 2, wherein the step of monitoring comprises monitoring objects within a class designated for monitoring.

6. The method according to claim 2, further comprising the step of performing the current stack walkbacks for the identified objects.

7. The method according to claim 6, further comprising the step of generating a statistics report including current stack walkbacks for the identified objects.

8. The method according to claim 7, further comprising the step of generating a web interface for user viewing of the statistics report at a computer display.

9. The method according to claim 1, wherein the software objects are Java objects.

10. A computer-implemented system for assisting the detection of critical memory leaks in a software program comprising:

means for monitoring an amount of available memory for the software program;

means for determining if the amount of available memory for the software program is less than a predetermined amount; and

means for, in response to determination, storing a current stack walkback of each object currently referenced by the software program prior to the available memory dropping below an amount necessary to store the current stack walkback wherein the current stack walkback assists in the defection of critical memory leak..

11. The system according to claim 10, further comprising:

means for monitoring a specified one or more analysis properties of the objects referenced by the software program, wherein the one or more specified analysis properties consists of at least one of an object's age and an object's instance count;

means for determining if any analysis property of the objects being referenced following a garbage collection process exceeds a predetermined limit for such analysis property, wherein the predetermined limit for an object's age is an object age limit and the predetermined limit for an object's instance count is an object instance count growth value; and

means for identifying software objects determined to have one or more analysis properties that exceeds that property's predetermined limit.

12. The system according to claim 11, further comprising means for calculating an object's age by timing a current period starting when the respective object was instantiated.

13. The system according to claim 11, further comprising means for calculating object instance count growth as the magnitude of growth of an object's instance count over a given time period.

14. The system according to claim 11, wherein means for monitoring comprises means for monitoring objects within a class designated for monitoring.

15. The system according to claim 11, further comprising means for performing the current stack walkbacks for the identified objects.

16. The system according to claim 17, further comprising means for generating a statistics report including current stack walkbacks for the identified objects.

17. The system according to claim 16, further comprising means for generating a web interface for user viewing of the statistics report at a computer display.

18. The system according to claim 10, wherein the software objects are Java objects.

19. An article of manufacture comprising machine-readable medium including program logic embedded therein for assisting the detection of critical memory leaks in a software program that causes control circuitry in a data processing system to perform the steps of:

monitoring an amount of available memory for the software program;

determining if the amount of available memory for the software program is less than a predetermined amount; and

in response to such determination, storing a current stack walkback of each object currently referenced by the software program prior to the available memory dropping below an amount necessary to store the current stack walkback wherein the current stack walkback assists in the detection of a critical memory leak.

20. The article of manufacture of Claim 19, further comprising the steps of:

monitoring a specified one or more analysis properties of the objects referenced by the software program, wherein the one or more specified analysis properties consists of at least one of an object's age and an object's instance count;

determining if any analysis property of the objects being referenced following a garbage collection process exceeds a predetermined limit for such analysis property, wherein the predetermined limit for an object's age is an object age limit and the predetermined limit for an object's instance count is an object instance count growth value; and

identifying any objects determined to have one or more analysis properties that exceeds that property's predetermined limit.

21. The article of manufacture of Claim 20, further comprising the step of calculating an object's age by timing a current period starting when the respective object was instantiated.

22. The article of manufacture of Claim 20, further comprising the step of calculating object instance count growth as the magnitude of growth of an object's instance count over a given time period.

23. The article of manufacture of Claim 20, wherein the step of monitoring comprises monitoring objects within a class designated for monitoring.

24. The article of manufacture of Claim 20, further comprising the step of performing the current stack walkbacks for the identified objects.

25. The article of manufacture of Claim 24, further comprising the step of generating a statistics report including current stack walkbacks for the identified objects.

26. The article of manufacture of Claim 25, further comprising the step of generating a web interface for user viewing of the statistics report at a computer display.

27. The article of manufacture of Claim 19, wherein the software objects are Java objects.